MORRISON & FOERSTER LLP
MICHAEL A. JACOBS (Bar No. 111664)
mjacobs@mofo.com
MARC DAVID PETERS (Bar No. 211725)
mdpeters@mofo.com
DANIEL P. MUINO (Bar No. 209624)
dmuino@mofo.com
755 Page Mill Road, Palo Alto, CA 94304-1018
Telephone: (650) 813-5600 / Facsimile: (650) 494-0792

BOIES, SCHILLER & FLEXNER LLP
DAVID BOIES (Admitted *Pro Hac Vice*)
dboies@bsfllp.com
333 Main Street, Armonk, NY 10504
Telephone: (914) 749-8200 / Facsimile: (914) 749-8300
STEVEN C. HOLTZMAN (Bar No. 144177)
sholtzman@bsfllp.com
1999 Harrison St., Suite 900, Oakland, CA 94612
Telephone: (510) 874-1000 / Facsimile: (510) 874-1460

ORACLE CORPORATION
DORIAN DALEY (Bar No. 129049)
dorian.daley@oracle.com
DEBORAH K. MILLER (Bar No. 95527)
deborah.miller@oracle.com
MATTHEW M. SARBORARIA (Bar No. 211600)
matthew.sarboraria@oracle.com
500 Oracle Parkway, Redwood City, CA 94065
Telephone: (650) 506-5200 / Facsimile: (650) 506-7114

Attorneys for Plaintiff
ORACLE AMERICA, INC.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

ORACLE AMERICA, INC. Plaintiff, v. GOOGLE INC. Defendant.	Case No. CV 10-03561 WHA **ORACLE AMERICA, INC.'S MOTION FOR JUDGMENT AS A MATTER OF LAW UNDER RULE 50(B) OR, IN THE ALTERNATIVE, FOR A NEW TRIAL** Date: July 26, 2012 Time: 8:00 a.m. Dept.: Courtroom 8, 19th Floor Judge: Honorable William H. Alsup

TABLE OF CONTENTS

TABLE OF CONTENTS
(continued)

Page

TABLE OF AUTHORITIES

PLEASE TAKE NOTICE that the following motion will be heard at 8:00 A.M. on July 26, 2012, or as soon thereafter as counsel may be heard, in Courtroom 8, 19th Floor, 450 Golden Gate Avenue, San Francisco, California before the Honorable William H. Alsup.

Plaintiff Oracle will and hereby does move for judgment as a matter of law under Rule 50(b), or in the alternative, for a new trial under Rule 59 as set forth more particularly in the Proposed Order filed concurrently with the motion. This motion is based on this Notice of Motion and Motion, the following Memorandum of Points and Authorities, documents incorporated by reference, the entire record in this action, any matters of which the Court may take judicial notice, and any evidence or argument presented at the hearing or on reply.

MEMORANDUM OF POINTS AND AUTHORITIES

INTRODUCTION

Google is liable for both copyright and patent infringement as a matter of law.

Oracle proved at trial that Google copied thousands of individual elements contained in 37 Java API packages and all of the intricate relationships among them. The witnesses for both parties agreed that designing these APIs took great creativity and skill.

The jury found Google infringed Oracle's copyrights in the structure, sequence and organization ("SSO") of the 37 API packages. The jury hung on the issue of whether Google proved its affirmative defense of fair use, but Oracle is entitled to judgment that Google's commercial, non-transformative and extensive copying of Oracle's copyrighted works does not constitute fair use as a matter of law.[1] Oracle is also entitled to judgment as a matter of law on Google's copying of the SSO of the Java documentation. Over Oracle's objection, the jury was instructed only to consider Google's copying of the English language descriptions contained in the documentation under a virtual identity standard. ECF No. 1018, Jury Instruction ("JI") 21, 24. But it is undisputed that the SSO of the Java and Android documentation is the same as the

[1] Oracle recognizes granting JMOL on some of the copyright issues would require the Court to overturn all or part of its order on copyrightability. Nonetheless, Oracle brings this motion to preserve its rights on appeal, particularly since the Court structured the trial to accommodate the possibility of the jury verdict being reinstated on appeal. *See* ECF No. 1202 at 2. Although not required to do so, Oracle also moves for JMOL on copyrightability out of an abundance of caution. *See* Section IE.1, *infra*.

SSO found in the code, which the jury found Google infringed. With proper instructions, a reasonable jury could only have found copyright infringement of the documentation as well. Oracle is further entitled to judgment based on Google's creation of a derivative work from the Java documentation, another issue that, over Oracle's objection, was not submitted to the jury.

Oracle also proved at trial that Google copied 11 individual code files. The jury found Google infringed its copyrights in one of those files and the Court granted Oracle's JMOL motion as to eight others. ECF No. 1123. Oracle directs this motion to the two remaining files, both of which contained comments Google admits were copied verbatim.

In addition, Google infringes United States Patents Nos. RE38,104 ("the '104 patent") and 6,061,520 ("the '520 patent"). Google infringes the '104 patent in two ways: the Resolve.c resolution functions that are part of the Dalvik Virtual Machine infringe Claims 11, 39, 40, and 41 and the dexopt tool that is also part of the Dalvik VM infringes Claims 27 and 29. Google infringes Claims 1 and 20 of the '520 patent through the operation of the dx tool, which is part of the Android SDK used by developers. Given the evidence in the record, Google's infringement can be determined as a matter of law, as no reasonable jury could find for Google.

The Court should also grant judgment on Google's defenses, as well as other affirmative defenses on which Google did not present any evidence or oppose Oracle's Rule 50(a) motion.

In the alternative, Oracle asks the Court to grant its motion for a new trial.

I. GOOGLE INFRINGED ORACLE'S JAVA-RELATED COPYRIGHTS

Oracle proved Google's copyright infringement by showing that (1) Oracle is the owner of the copyrighted works and (2) Google copied protected elements from those works. ECF No. 1018, JI 23-24. *Feist Publ'ns, Inc. v. Rural Tel. Serv. Co.*, 499 U.S. 340, 361 (1991).

A. Oracle Owns The Asserted Copyrights

By Google's choice, copyright ownership was not posed to the jury. ECF No. 1018, JI 23; RT 2392:10-2396:6. Google elected instead to present the issue as a question of law for the Court. RT 2392:12-2394:14. The Court denied Google's motion that it was entitled to JMOL as to ownership. *See* ECF No. 1165. Oracle accordingly believes no issue as to ownership remains.

Nonetheless, to be prudent, Oracle requests that the Court grant Oracle judgment as to ownership of the asserted works. As the Court found in denying Google's JMOL motion, Oracle presented evidence that it was the owner and copyright holder of the 37 API packages and the eleven code files. ECF No. 1165 at 1-2. *See also* TX 476, RT 2233:1-17, 2239:16-23 (Reinhold).

"A certificate of registration raises the presumption of copyright validity and ownership." *Dream Games of Ariz., Inc. v. PC Onsite*, 561 F.3d 983, 987 n.2 (9th Cir. 2009); *see also* 17 U.S.C. § 410(c). "This presumption of ownership is true even for individual works that are broadly registered as part of a compilation or derivative work." ECF No. 1165 at 2 (citing *United Fabrics Int'l, Inc. v. C&J Wear, Inc.*, 630 F.3d 1255, 1257-59 (9th Cir. 2011)). Google never presented any evidence showing Oracle is not the owner of the copyrighted works, and the Court found the issue of whether Oracle owns the works was a question of fact for the jury, which Google waived its right to contest by electing not to submit it to the jury. ECF No. 1165 at 2.

Oracle also incorporates by reference here its proposed findings of fact and conclusions of law and prior briefing on this issue. *See* ECF No. 1049 ¶¶ 1-8, 142-45; ECF No. 1093 at 2-8.

B. Google Infringed By Copying Comments From Oracle Source Code

There is no dispute that Google copied comments from two Java files Google admits the "comments came from the copyrighted material." ECF No. 1018, JI 27. A comparison of Oracle's CodeSource.java file (TX 623.9) to Android's CodeSourceTest.java file (TX 1039) shows that, except for some HTML commands, the copied comments are "syntactically . . . identical." RT 1262:13-1263:4 (Mitchell). Google also copied comments identically from the CollectionCertStoreParameters.java file. *Compare* TX 623.10 *with* TX 1040; *see* RT 1253:9-10 (Mitchell). The copied comments are quantitatively significant: they amount to about 25% of one Oracle file (TX 623.10), and about 2.90% of the other (TX 623.9).

Google's verbatim copying is immediately recognizable. No reasonable jury could find it was *de minimis*. Copying can only be *de minimis* "if it is so meager and fragmentary that [compared to the work as a whole] the average audience would not recognize the appropriation." ECF No. 1018, JI 28; *Fisher v. Dees*, 794 F.2d 432, 434 n.2 (9th Cir. 1986). The extent of the copying "is measured by considering the qualitative and quantitative significance of the copied portion in relation to the plaintiff's work as a whole." *Newton v. Diamond*, 388 F.3d 1189, 1195

(9th Cir. 2004). Even if the copied material is a "quantitatively very small part" of the work as a whole, "[t]he smallness alone is not enough by itself to avoid liability." *Mktg. Tech. Solutions, Inc. v. Medizine LLC*, 2010 U.S. Dist. LEXIS 50027, at *10 (S.D.N.Y. Apr. 23, 2010).

C. Google Copied Java Specifications Into Android Specifications

Over Oracle's objection, the Court submitted to the jury only the question of whether Google infringed Oracle's copyrights in its documentation by copying the English-language descriptions of API elements. *See* ECF No. 1018, JI 21; ECF No. 997 at 1-3 (Oracle written objections); RT 2833:15-2389:7 (charging conference). The Court did not submit the issue of whether the SSO of the 37 API packages was also copied into the Android specifications. Because the jury found Google infringed the SSO of the 37 API packages, a reasonable jury would have found Google infringed the indisputably identical SSO in the documentation as well.

The selection, arrangement and structure of documentation is protectable. *Urantia Found. v. Maaherra*, 114 F.3d 955, 959 (9th Cir. 1997); *see also Jacobsen v. Katzer*, 2009 U.S. Dist. LEXIS 115204, at *9-10 (N.D. Cal. Dec. 10, 2009) ("selection, ordering and arrangement" of text files reflecting decoder information from model railroad manufacturers). This is true even if the individual elements are not protectable. *Metcalf v. Bochco*, 294 F.3d 1069, 1074 (9th Cir. 2002) ("The particular sequence in which an author strings a significant number of unprotectable elements can itself be a protectable element. Each note in a scale, for example, is not protectable, but a pattern of notes in a tune may earn copyright protection.").

In *Baker v. Selden*, the Supreme Court found the detailed description in a document of a system is copyrightable even if the underlying system it describes is not. *Baker v. Selden*, 101 U.S. 99 (1879). Here, unlike *Baker*, the structure described by the Java API documentation is itself copyrightable because it is the structure of a computer program. *Johnson Controls, Inc. v. Phoenix Control Sys., Inc.*, 886 F.2d 1173, 1175 (9th Cir. 1989). Moreover, the Ninth Circuit treats *Baker* as a "blank forms" case and holds that when such forms are integrated with text to convey information they are copyrightable. *See, e.g., Edwin K. Williams & Co. v. Edwin K. Williams & Co.-East*, 542 F.2d 1053, 1060-61 (9th Cir. 1976). In any event, there is no question under *Baker* that the documentation is copyrightable. That includes its original and creative SSO.

The parties agree the selection and structure expressed in the documentation is the same as the selection and structure expressed in the source code. *See* RT 606:14-608:3 (Reinhold); ECF No. 1043 at 14 (Google JMOL). The jury found Google infringed the "overall structure, sequence and organization of the copyrighted works." ECF No. 1089 ¶ 1. This finding was well supported. Experts for both parties testified the SSO of the 37 API packages in Android and Java is virtually identical. *See* RT 1244:17-1246:3 (Mitchell); RT 2214:3-9 (Astrachan) (SSO is "virtually identical"). Android developer Bob Lee conceded this too. RT 1174:9-12 (Lee). A reasonable jury could only have found Google infringed the SSO in the documentation as well.

This question was never put to the jury, however. Instead, at Google's request, the jury was asked to compare only the English language descriptions. Even so, a properly instructed jury would have found in Oracle's favor. Over Oracle's objection, the jury was asked to determine infringement based on whether the English language descriptions were "virtually identical" in Java and Android rather than "substantially similar." *See* ECF No. 1018 JI 24; ECF No. 997 at 3 (written objections); RT 2399:12-2400:5 (charging conference). Lead Android developer Bob Lee admitted the English-language descriptions within the Android specifications were paraphrased from Sun's specifications and were therefore "substantially similar." RT 1191:4-13, 1175:25-1176:3 (Lee). Mr. Lee was shown three examples of this paraphrasing and acknowledged the same level of similarity exists across the full documentation for the 37 Java API packages. RT 1175:25-1176:3 (Lee); TX 610.2, 767.

This admitted paraphrasing warranted judgment in Oracle's favor of copying of the English-language descriptions even under the virtual identity standard. But those descriptions should have been considered together with, and in the context of, the extensively copied selection, structure, sequence and organization of the API elements, even if they were not individually protectable. *See, e.g., Three Boys Music Corp. v. Bolton*, 212 F.3d 477, 485 (9th Cir. 2000) ("It is well settled that a jury may find a combination of unprotect[a]ble elements to be protect[a]ble ... because 'the over-all impact and effect indicate substantial appropriation.'") (citation omitted). For roughly 11,000 pages, the 37 Android packages are laid out in the same organization, with thousands of the same names, nearly all the same packages, classes and methods with the same relationships among all elements, and a paraphrased English definition next to each. The Court should have applied the substantial similarity standard to this range of creative expression. But Oracle is entitled to JMOL under either a substantial similarity or virtual identity standard.

D. Google's Copying Is Not Fair Use

No reasonable juror could find based on the trial evidence that Google met its burden to prove fair use, and Oracle is entitled to judgment as a matter of law on this defense.

1. Google's Use of the Copyrighted Work Is Purely Commercial

a. Google's Use Is Commercial

"Although not controlling, the fact that a new use is commercial as opposed to non-profit weighs against a finding of fair use." *Elvis Presley Enters. Inc. v. Passport Video*, 349 F.3d 622, 627 (9th Cir. 2003), *overruled on different grounds in Flexible Lifeline Sys. v. Precision Lift, Inc.*, 654 F.3d 989 (9th Cir. 2011).

Android is hugely profitable. *See* RT 1458:12-16; 1456:15-19 (Schmidt); RT 2225:18-2226:24 (Agarwal). Google distributes Android to increase use of Google services, which generate advertising revenue. RT 1458:12-16 (Schmidt). Google documents describe Android as a "critical" platform for five Google business units, and a $10 billion opportunity. TX 431 at 3.

Google argues that it only profits *indirectly* from Android because it distributes the software free of charge. *See* ECF No. 1092 at 8. This does not matter. Google's use is clearly commercial. "The crux of the profit/nonprofit distinction is not whether the sole motive of the use is monetary gain but whether the user stands to profit from exploitation of the copyrighted material without paying the customary price." *Harper & Row Publishers, Inc. v. Nation Enters.*, 471 U.S. 539, 562 (1985). The first factor weighs strongly against fair use.

b. Google's Use Is Not Transformative

"A use is considered transformative only where a defendant changes a plaintiff's copyrighted work or uses the plaintiff's copyrighted work in a different context such that the plaintiff's work is transformed into a new creation." *Wall Data Inc. v. L.A. Cnty. Sheriff's Dep't*, 447 F.3d 769, 778 (9th Cir. 2006). "In cases where 'use is for the same intrinsic purpose as [the copyright holder's] . . . such use seriously weakens a claimed fair use.'" *Id.* (quoting *Worldwide*

Church of God v. Philadelphia Church of God, Inc., 227 F.3d 1110, 1117 (9th Cir. 2000)). The Supreme Court explains: "The enquiry here may be guided by the examples given in the preamble to § 107, looking to whether the use is for criticism, or comment, or news reporting, and the like, see § 107." *Campbell v. Acuff-Rose Music, Inc.*, 510 U.S. 569, 578-79 (1994) (citations omitted).

In *Leadsinger, Inc. v. BMG Music Publ'g*, the Ninth Circuit rejected the defendant's claim that its karaoke machine was transformative, noting, "It is reasonable to infer that Leadsinger does not add to or alter the copyrighted lyrics, which would undermine the device's ability to enable consumers to sing along with the recorded music." 512 F.3d 522, 530 (9th Cir. 2008). Ninth Circuit cases following *Campbell* have found transformative use where the accused infringer's work "served an entirely different function" from the original. *See Kelly v. Arriba Soft Corp.*, 336 F.3d 811, 818 (9th Cir. 2003). This is not the case here. Oracle objected to giving the jury an instruction on transformative use. *See, e.g.*, ECF No. 1005 at 4-8.

Unlike the parody in *Campbell*, Google's use of the copied materials in Android is nothing like "the examples given in the preamble to § 107." *See* 17 U.S.C. § 107. Google's purpose for the Java APIs was the same as Oracle's: to attract developers by providing them with pre-written libraries of reusable code. *See* RT 584:10-585:5 (Reinhold); RT 1783:15-22 (Bornstein) ("The goal of the project was to provide something that was familiar to developers"). Google's argument that it "transformed" Java by creating the first full smartphone stack to use the Java APIs is simply false. Oracle licenses the Java APIs in its Java ME products for exactly this purpose. Smartphones including the RIM Blackberry, the Danger Sidekick/Hiptop, and the Nokia Series 60 incorporate Oracle's licensed technology. RT 959:20-23 (Swetland); 1585:21-23 (Rubin); 300:18-19 (Ellison); 383:6-9 (Kurian); 1102:3-10 (Cizek); 1922:22-25 (Gering).

The difference between Android and these other platforms is not expressive transformation but business strategy. If Google's position were accepted, the idea of "transformation" would severely undermine copyright protection: anyone claiming to have a better business model for distributing the copyrighted work would be able to claim "fair use." That Google licenses Android under the Apache "open source" license while Oracle licenses Java both commercially and under a different open source license—the GPL—"transforms" nothing.

2. The Copyrighted Work Is Creative In Nature

Copyright law specifically protects computer programs. 17 U.S.C. § 101. The Ninth Circuit has held that while "software products are not purely creative works, copyright law nonetheless protects computer software." *Wall Data*, 447 F.3d at 780. In that case, the court found the "nature of the work" factor weighed against fair use where the software products "were developed over several years, and required a multi-million dollar investment…." *Id.* Oracle presented undisputed evidence of both at trial. *See, e.g.*, RT 687:21-688:24 (Reinhold).

Moreover, witnesses from both sides testified that designing APIs was a creative endeavor. *See, e.g.*, RT 513:12-18; 513:21-514:12; 515:14-23 (Screven); 627:21-628:1 (Reinhold); 741:9-742:3; 747:5-9; 748:7-13; 752:5-14; 831:17-832:4 (Bloch); 1220:6-12; 1238:11-1239:12; 1240:16-20 (Mitchell); 1775:3-16 (Bornstein); RT 2209:7-8 (Astrachan); TX 1090 (Astrachan Dep.). This factor also weighs against fair use.

3. Google Uses Valuable, Core Portions of the Copyrighted Work

In analyzing the third factor, courts look at the quantitative and qualitative significance of the material taken in relation to the plaintiff's work. *Harper & Row*, 471 U.S. at 564-65 (1985) (finding infringement based on copying of 300 words of Gerald Ford's memoirs).

Google engineers testified they selectively copied what they thought were the best APIs for a mobile platform, the "good stuff" from Java. *See* RT 1782:6-1785:4 (Bornstein); RT 981:22-982:21 (Lee); TX 1067 (Lee Dep.). But unlike *Harper & Row*, Google copied thousands of elements from the 37 API packages and their entire SSO. RT 1248:11-1249:25 (Mitchell); RT 2191:9-2192:3 (Astrachan). Google's expert agreed these declarations would "replicate every structural and organizational element" of the 37 packages in suit. RT 2191:17-20 (Astrachan). Google copied not just the heart of Oracle's work, but its spine and much of its skeletal structure.

Google's argument that it copied only what was necessary for compatibility is legally and factually incorrect. When an intended use is commercial, courts give little weight to the claim that a defendant only copied what was necessary for its intended use. In *Elvis Presley*, the Ninth Circuit found no fair use for television performance excerpts in an Elvis biography, despite acknowledging that "[i]t would be impossible to produce a biography of Elvis without showing some of his most famous television appearances for reference purposes." 349 F.3d at 629. That Google added its own implementation of the method bodies is no excuse. "[N]o plagiarist can excuse the wrong by showing how much of his work he did not pirate." *Harper & Row*, 471 U.S. at 565 (quoting *Sheldon v. Metro-Goldwyn Pictures Corp.*, 81 F.2d 49, 56 (2d Cir. 1936)).

4. Google's Use Harms The Potential Market For And Value Of The Copyrighted Work

Courts balance the first and the fourth fair use factors. "[I]f the purpose of the new work is commercial in nature, 'the likelihood [of market harm] may be presumed.'" *Elvis Presley*, 349 F.3d at 631 (quoting *A&M Records, Inc. v. Napster, Inc.*, 239 F.3d 1004, 1016 (9th Cir. 2001)). In assessing this factor, it is necessary to "consider not only the extent of market harm caused by the particular actions of the alleged infringer, but also 'whether unrestricted and widespread conduct of the sort engaged in by the defendant . . . would result in a substantially adverse impact on the potential market' for the original." *Campbell*, 510 U.S. at 590. Courts look not at harm to the market for potential derivative works as well. *See id.* at 592-93.

Android's infringement has substantially harmed the actual and potential market for Oracle's Java mobile products. There are 750,000 Android-compatible device activations each day, and each of those devices contains the 37 API packages from Java. RT 1017:4-16 (Morrill). Android phones compete directly with Java smart phones (such as the RIM Blackberry). RT 1922:22-25 (Gering); *see also* RT 2062:5-12 (McNealy). Furthermore, Google has fragmented Java and undercut its "write once, run anywhere" promise. TX 172 (email from Bornstein to Rubin describing Android as a "fork" of Java); RT 2287:13-2288:5 (Mitchell); *see also* RT 984:22-24; 981:19-21 (Lee); 1010:1-7 (Morrill) (Android is not Java compatible). Android is diverting licensing revenue to which Oracle is entitled for its Java mobile products. This factor also weighs strongly against fair use.

E. Judgment Should Be Granted In Oracle's Favor On Issues That Were Not Presented To The Jury

1. Oracle Is Entitled To Judgment As A Matter Of Law On Copyrightability

The issue of copyrightability of the 37 API packages was determined by the Court, not the jury. Accordingly, no motion for JMOL is required. *See* Fed R. Civ. P. 50(a)-(b); *Granite State Ins. Co. v. Smart Modular Techs., Inc.*, 76 F.3d 1023, 1030-31 (9th Cir. 1996) (Rule 52 governed equitable estoppel claim tried to court because "Rule 50(a) applies only to issues tried by a jury"); 9 James W. Moore et al., *Moore's Federal Practice* § 50.05[1] (3d ed. 2012). Nonetheless, out of an abundance of caution, Oracle requests JMOL on copyrightability here. Oracle incorporates by reference its proposed findings of fact and conclusions of law on copyrightability, and its response to Google's proposed findings. *See* ECF No. 1049 ¶¶ 1-57, 142-165; ECF No. 1081 at 1-16, 37-55. Oracle also incorporates by reference the prior briefing on copyrightability. This includes ECF Nos. 339, 611, 780, 824, 833, 853, 859, 900, 956, 986, 1118, 1138, 1191 and 1197.

The evidence presented at trial on copyrightability was overwhelming. As noted in section I.D.2 above, fact and expert witnesses from both sides testified that designing APIs is a creative and challenging task. Nobody testified to the contrary.

The API packages themselves are expressed in a detailed and complex structure, with many hierarchies and interdependencies. These were illustrated in part in TX 1028, the Java API package poster used by developers when programming for J2SE version 5.0. This poster reflects only the high level class and interface relationships for some of the API packages in version 5.0. RT 599:15-600:3 (Reinhold). The types of relationships shown at trial included the following: (1) classes can have one or more subclasses, each of which inherits the methods and fields of the classes above it in the hierarchy (RT 1225:10-16 (Mitchell)); (2) interfaces are used to relate different classes that share common characteristics (RT 589:13-17, 590:5-23, 601:22-25 (Reinhold)); (3) methods can contain parameters that are defined in other classes located within, or outside, the package in which the method is found (RT 1239:24-1240:8 (Mitchell)); (4) classes and subclasses can be contained within the hierarchy of one package but defined in another (RT 601:14-24 (Reinhold)); (5) interfaces are often arranged hierarchically in a manner similar to classes (RT 1219:14-23 (Mitchell)).

The detailed expression of this structure cannot possibly be just an idea, as Google has sometimes claimed. Nor is it driven or constrained by function. Very little structure is required for the APIs to operate with the virtual machine or computer. If function were the only concern, all of the classes could have been placed in a one giant package. RT 619:13-23 (Reinhold). A primary purpose of the structure, sequence and organization of the APIs is to make them easy to learn and easy for developers to use. RT 619:24-620:6 (Reinhold); RT 741:2-742:2 (Bloch); TX 624 at 4. Aesthetics matter. RT 752:5-14 (Bloch).

The many creative choices exercised by API designers extend not just to the selection and structure of classes and methods to carry out a given API, but also to the decision whether to include a particular API package in the library in the first place. There is no requirement that any particular API be included, or that any specific method or class be included within that API. The Java API packages have grown dramatically, from the seven API packages that were included in the first release, to the 166 packages included with version 5.0, to 209 packages included with version 7.0. RT 631:19-25 (Reinhold). The individual API packages themselves have also significantly expanded over time. *See, e.g.,* RT 1243:13-1244:16 (Mitchell) (growth of java.util). *Compare* TX 2564 at 615 *with* TX 610.2 (showing java.util had only 10 classes and 113 methods in 1996 compared to 49 classes and 762 methods in version 5.0). Dr. Reinhold testified that Sun and Oracle did not have to create so many Java API packages, but did so "in order to – to encourage the adoption of the Java platform by adding more and more facilities to make it an attractive platform for developers to use." *Id.* at RT 632:1-6 (Reinhold). Other software platforms, like C, have much less extensive APIs. *See id.* at RT 632:7-20 (Reinhold).

The Court should also grant JMOL for Oracle on the copyrightability of the selection and arrangement of the names in the 37 Java APIs. Both parties testified that API designers thoughtfully selected thousands of names for aesthetic purposes and consistency. RT 628:2-21 (Reinhold); TX 624 (Bloch presentation) at 17 ("Code should read like prose."); RT 746:20-748:13 (Bloch). The names are organized within the same complex and creative structure as the API elements they label. *See Lamps Plus, Inc. v. Seattle Lighting Fixture Co.*, 345 F.3d 1140, 1147 (9th Cir. 2003) (combination of unprotectable elements is eligible for copyright protection "if those elements are numerous enough and their selection and arrangement original enough that their combination constitutes an original work of authorship."). Many cases have held the selection and arrangement of individually unprotectable elements within a software program, including names, can be copyrightable. *See, e.g.*, *Merch. Transaction Sys., Inc. v. Nelcela, Inc.*, 2009 U.S. Dist. LEXIS 25663, at *58 (D. Ariz. Mar. 17, 2009); *see also Health Grades, Inc. v. Robert Wood Johnson Univ. Hosp., Inc.*, 634 F. Supp. 2d 1226, 1238 (D. Colo. 2009) (declining to find short phrase that was original expression uncopyrightable).

Google did not establish at trial that the doctrines of merger and *scenes a faire* apply to bar copyrightability of APIs. The Court warned Google on summary judgment that it would have to present evidence as to each specific element of the APIs it contended was unprotectable. ECF No. 433 at 9. Google did not present sufficient evidence at trial to establish any particular method declaration was a *scene a faire* or was the only possible way to express a given function.

Nor could it have. It is apparent, given the complexity of the structure and the many possibilities for selection, that there are countless ways to design and express the Java API packages, so the doctrine of merger does not apply. *See Satava v. Lowry*, 323 F.3d 805, 812 n.5 (9th Cir. 2003) ("Under the merger doctrine, courts will not protect a copyrighted work from infringement if the idea underlying the copyrighted work can be expressed in only one way, lest there be a monopoly on the underlying idea"). Dr. Reinhold testified that "In anything except the most trivial API design, there are so many choices to be made I wouldn't even know how to start counting them." RT 627:21-628:1 (Reinhold); *see also id.* at 2228:2-16 (discussing complexity of java.nio design). Professor Mitchell agreed, emphasizing that API design starts with a "clean slate." RT 1240:9-20. It took almost two years to design the APIs for java.nio and its related sub-packages, and the specification went through over 30 separate drafts. RT 623:17-626:13, 627:21-629:6 (Reinhold). No Google witness disputed any of these points.

Google also did not put on any evidence at trial sufficient to establish *scenes a faire*. "Under the *scenes a faire* doctrine, when certain commonplace expressions are indispensable and naturally associated with the treatment of a given idea, those expressions are treated like ideas and therefore not protected by copyright." *Swirsky v. Carey*, 376 F.3d 841, 850 (9th Cir. 2004). The evidence showed APIs solving the same kinds of problems can be designed very differently. Dr. Reinhold gave the example of the java.util.logging API package, contrasting it with a competing open source Java logging API called Log4J. RT 630:11-631:18 (Reinhold). Dr.

Mitchell discussed how data collections are handled in different ways in APIs in Java, C++ and Smalltalk, and how even within Java, the design of the Java.util package has changed significantly over time. RT 1240:23-1244:16 (Mitchell).

It is undisputed that Google could have written its own different APIs to provide the functionality of the Java APIs. RT 2213:8-10 (Astrachan). This is proven by the fact that Google did write many of its own APIs when it wanted to. RT 2213:17-19 (Astrachan). Even at the individual method level, Google could have used different method names, different parameter names, a different parameter sequence, or thrown exceptions in a different order or not at all and still accomplished the same task. *See, e.g.*, RT 1249:2-12 (Mitchell); ECF No. 1191 at 2-3; ECF No. 1118 at 12-13; TX 984 at 302-04. But this case was never about any one individual method or group of methods. It was about Google's nearly identical copying of the selection, sequence, structure and organization of thousands of methods and other API elements. Even if Google were correct that the declaration of an individual method is uncopyrightable, Google selected and implemented thousands of Oracle's declarations in the same SSO in which Oracle wrote them. "[A] claim of copyright infringement can be based on infringement of a combination of unprotected elements." *Dream Games*, 561 F.3d at 988. In addition, even if Google could prove merger, it would still be liable for its "nearly identical copying." *Apple Computer, Inc. v. Microsoft Corp.*, 35 F.3d 1435, 1444 (9th Cir. 1994).

Google also contended at trial that the 37 API packages are simply a "functional requirement for compatibility." This is not the correct legal standard. Google relies on *Sega* and *Sony*, both of which are reverse engineering fair use cases with clearly distinguishable facts.

Google has also not shown the Java APIs are merely "functional requirements." All computer programs are functional, as is the source code that expresses them. Claiming something is functional, without more, says nothing. The Ninth Circuit did not conduct a detailed analysis of this issue in *Sega* because the question of infringement in the final product was reserved by Sega and left for remand. *See Sega Enters. Ltd. v. Accolade, Inc.*, 977 F.2d 1510, 1528 (9th Cir. 1993). But the decision shows that in determining whether an element of a computer program is a mere functional requirement the court will look to the level of creative expression involved.

That is what the court used to distinguish the S-E-G-A 20 byte initialization code from the "*original program*" in *Atari v. Nintendo*, 975 F.2d 832, 840 (Fed. Cir. 1992). *Sega*, 977 F.2d at 1524 n.7 (emphasis in original). As discussed above, the API package designs are highly expressive, not merely functional, and their design is important to developer comprehension.

Similarly, in *Sony*, the Ninth Circuit and district court opinions both emphasize that Sony did not accuse the final product of infringement. *See, e.g., Sony Computer Entm't Inc. v. Connectix Corp.*, 48 F. Supp. 2d 1212, 1217 (N.D. Cal. 1999) (*"Sony I"*) ("Sony's copyright infringement claim is based on a theory of *intermediate* infringement.") (emphasis added); *Sony Computer Entm't, Inc. v. Connectix Corp.*, 203 F.3d 596, 904 (9th Cir. 2000) ("*Sony II*") ("nor does Sony contend that Connectix's final product contains infringing material"). As a result, neither decision analyzes what was copied in the final product at all, and there is no indication in the opinions that what was copied contained creative expression or was similar to the APIs here.

Google also uses the term "compatibility" very differently from the Ninth Circuit in *Sega*. The API packages are not like a hardware interface that Google had to adopt if it wanted to use the Java programming language. As noted above, Google designed many of its own API packages, and the experts agreed that Google could have designed its own corresponding 37 API packages if it wanted to. Only about 60 classes must be present in the APIs for the Java language to function, and for most of these there is no requirement that the class contain any particular method or methods — the language simply expects that a class by that name will exist. RT 684:16-685:2 (Reinhold); RT 2196:1-4 (Astrachan); TX 1062; TX 984. Google copied far more. It also copied far more than would have been required for compatibility with the APIs, copying, for example, thousands of parameter names and throws clauses. *See* RT 1248:11-1249:12 (Mitchell); ECF No. 1191 at 3.

Moreover, the evidence showed Android is not compatible with Java. *See, e.g.,* RT 1007:6-11 (Morrill) (Android not compatible); TX 383 at 8 (Android FAQs); RT 1331:16-1332:2 (Mitchell); RT 2221:11-2222:3 (Astrachan) (entry point must be changed); RT 2287:9-22 (Mitchell) (bytecode); ECF No. 1118 at 18-19 (listing example categories of Java applications that will not run on Android). Dan Bornstein testified that achieving compatibility was not even a goal for Google. RT 1783:15-22. And Google presented no evidence that copying the 37 API packages allowed the meaningful re-use of any significant amount of pre-existing code. By implementing a partial version of Java, Google harmed compatibility, not furthered it.

Lastly, Google did not present sufficient evidence to show the APIs are a "method of operation" or "system" under section 102(b). The sweeping approach that "methods of operation" are uncopyrightable taken by the First Circuit in *Lotus v. Borland* has never been adopted by any other circuit. The Tenth Circuit expressly declined to follow *Lotus*. *Mitel, Inc. v. Iqtel, Inc.*, 124 F.3d 1366, 1372 (10th Cir. 1997). In determining whether the nonliteral components of a program are protectable, the Ninth Circuit looks at "whether, on the particular facts of each case, the component in question qualifies an the expression of an idea, or an idea itself." *Johnson Controls*, 886 F.2d at 1175.

Lotus also incorrectly defines "method of operation" as "a means by which a person operates something." This is very close to the definition of computer program under the Copyright Act, which is "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result." 17 U.S.C. § 101. It threatens to swallow the rule whole. The Java APIs also do not fit the *Lotus* definition of "method of operation" because they are not "a means by which a person operates something." The *Lotus* court viewed the Lotus 1-2-3 menu command hierarchy as a "method of operation" because the commands were the actual keystrokes that a person would type to use the Lotus 1-2-3 program. *Lotus Dev. Corp v. Borland Int'l, Inc.*, 49 F.3d 807, 809 (1st Cir. 1995), *aff'd by an evenly divided court*, 516 U.S. 233 (1996); *see id.* at 815. Those are not the facts here. On a computer or an Android phone, there is no key or touchscreen menu labeled "HandshakeCompletedEvent," or "getCipherSuite()," or anything corresponding to the thousands of other Java API elements.

Google also never proved the APIs are a "system." It never even defined what it meant by "system" at trial. But even if the APIs could be labeled a system at a higher level of abstraction, their particular expression is still protected by copyright. *Toro Co. v. R&R Prods. Co.*, 787 F.2d 1208, 1212 (8th Cir. 1986) (expression of parts numbering system copyrightable if original).

2. Google Created An Infringing Derivative Work

The Android source code for the 37 API packages was derived from Oracle's copyrighted API specifications. Former Android engineer Bob Lee admitted Google consulted the Java API specifications when developing Android. RT 982:25-983:3 (Lee); *see also* RT 981:7-21 (Lee). Google's outside contractor, Noser, was hired to implement core libraries based on the Java API specifications. RT 985:3-6 (Lee). Dan Bornstein confirmed his team used the Java specifications to derive information for implementing the APIs in Android. RT 1836:19-1837:2 (Bornstein).

A "derivative work" is defined as "a work based upon one or more preexisting works, such as a translation . . . or any other form in which a work may be recast, transformed, or adapted." 17 U.S.C. § 101. Ninth Circuit law supports Oracle's derivative works claim. *See, e.g., Micro Star v. Formgen Inc.*, 154 F.3d 1107, 1112 (9th Cir. 1998) (seller of new levels for video game infringed by copying "story" of plaintiff's video game even though it did not copy computer art files); s*ee also Twin Peaks Prods., Inc. v. Publ'ns Int'l, Ltd.*, 996 F.2d 1366, 1373-74 (2d Cir. 1993) (detailed recounting of plot elements of television series was infringement).

The Java API specifications are the detailed description of the class libraries. They are like a detailed plot outline, and are just as protectable. *See Sheldon*, 81 F.2d at 55-56 ("The play is the sequence of the confluents of all these means, bound together in an inseparable unity; *it may often be most effectively pirated by leaving out the speech, for which a substitute can be found, which keeps the whole dramatic meaning.*") (emphasis added); *see also Twentieth Century-Fox Film Corp. v. MCA, Inc.*, 715 F.2d 1327 (9th Cir. 1983) (reversing decision as to whether plots of Star Wars and Battlestar Galactica were sufficiently similar to support infringement claim). Courts have applied the reasoning in *Sheldon* to cases involving computer programs. *See, e.g., eScholar, LLC v. Otis Educ. Sys., Inc.*, 2005 U.S. Dist. LEXIS 40727, at *25 (S.D.N.Y. Nov. 3, 2005) (comparing copyright protection of structure of computer program to plot elements in *Sheldon*). *Micro Star* applied a similar analysis. 154 F.3d at 1112.

Likewise, in *SAS Inst., Inc. v. S&H Computer Sys., Inc.*, the court held defendant created an infringing derivative work that was "based on" the SAS software by copying its structure even though there were relatively few examples of line-by-line copying, emphasizing that: "to the extent that it represents copying of the organization and structural details of SAS, *such copying pervades the entire S&H product*." 605 F. Supp. 816, 830 (M.D. Tenn. 1985) (emphasis added).

The reasoning of these cases applies equally to Google's copying here. But Google's copying was much more extensive. Google expert Owen Astrachan conceded that the Android source code was "based on the specification" (RT 2219:7-18 (Astrachan)), and that the copied method declarations in Android are like the "sub-sub-sub-chapter headings" in the SSO. *Id.* at 2215:2-5; *see also* RT 1253:16-18 (Mitchell) ("[T]he narrative is reflected in the source code because the source code is a program that in a sense carries out that narrative, does what the explanation requires for this method."). As in *SAS*, Google's deliberate copying of the SSO "pervades the entire" 37 Android API packages. *SAS*, 605 F. Supp. at 830. Over Oracle's objection, the jury was not given an instruction on the creation of a derivative work in the Android code from the Java documentation. *See* RT 2434:2-2435:17 (charging conference).

II. GOOGLE INFRINGED THE ASSERTED CLAIMS OF THE '104 PATENT

A. Android's Resolve.c infringes claims 11, 39, 40, and 41 of the '104 patent because Dalvik bytecode instructions contain symbolic references

The only dispute as to infringement by Android's Resolve.c is whether Dalvik bytecode instructions contain "symbolic references." RT 4106:21-22 (Jacobs); RT 4154:6-11 (Van Nest).

1. A field index is a symbolic reference that is contained in a Dalvik bytecode instruction

A field index in a Dalvik bytecode instruction meets the Court's definition of "symbolic reference." The Court construed the term "symbolic reference" as "a reference that identifies data by a name other than the numeric memory location of the data, and that is resolved dynamically rather than statically." ECF No. 137 at 22.

A field index—also called field@CCCC generally or "01" in specific examples of field indices in the trial testimony—is a reference to data to be obtained in accordance with a corresponding numerical reference, and identifies that data by a name other than the numeric memory location of the data. RT 3228:14-3229:25 (McFadden); RT 3303:2-3304:20 (Mitchell). In order to obtain data from the data object containing the value of the field, the Dalvik VM uses the resolver functions of Resolve.c to resolve the field index to a numeric memory location that is then used to obtain the value. RT 3646:24-3647:25 (McFadden); RT 3308:18-3309:24 (Mitchell). The Dalvik VM resolves type indices, method indices, and string indices in much the same way as field indices, and these indices are symbolic references for the same reason. *See* RT 3239:17-21 (McFadden); 3310:4-3311:1 (Mitchell); TX 736 at 2.

The Dalvik bytecode instruction that was the focus of both parties' evidence and argument is the IGET instruction, which corresponds to the "LOAD 'y'" instruction in the '104 patent. RT 3297:10-3302:2 (Mitchell); RT 3956:2-3961:6 (August). The IGET instruction (together with the IPUT instruction) "[performs] the identified object instance field operation with the *identified field*, loading or storing into the value register." TX 735 at 6 (emphasis added). The IGET instruction contains three operands—vA, vB, and field@CCCC—where the third operand field@CCCC is the field index. TX 735 at 6; RT 3221:8-10 (McFadden). The field index in the IGET instruction identifies the field from which the data is to be obtained by IGET. Mr. McFadden testified:

> Q. Can you explain what the iget instruction is?
>
> A. That is the instance field get instruction. What that means is there is an object somewhere and you need to get a piece of data out of it. The data is stored in fields. So what this instruction does is it finds the instance of the object and retrieves the data from the specified field.

RT 3221:2-7 (McFadden); *see also* RT 3968:10-15 (August). Thus even Google's witness's testimony established that the "data" that the IGET instruction specifies and retrieves is *data from the instance of an object*.

Every Google witness confirmed the field index contained in Dalvik's IGET instruction is not the numeric memory location of the value of the data from the instance of an object. Messrs. McFadden and Bornstein testified it was not. RT 3614:22-3615:16 (Bornstein); 3761:19-3762:6 (McFadden). Google's expert Dr. August likewise testified likewise. RT 3970:20-3971:3 (August). Indeed, "the Dalvik IGET instruction ***never*** contains the numerical memory location of the actual field data that it is supposed to get." RT 3761:14-18 (McFadden) (emphasis added).

Accordingly, no reasonable jury could conclude that the field index in a Dalvik IGET instruction is not a symbolic reference to the "actual field data that it is supposed to get." Under the Court's construction, for the field index in the IGET instruction to be a symbolic reference, it is enough that it identifies—"specifies," in Mr. McFadden's words—data to be obtained, by something other than the data's location.

2. According to the '104 patent, the "data" that the claimed symbolic references refer to are actual data in a data object

That data from an instance object is the "data" that the claimed symbolic reference refers to, and not some other data, follows from the claim language of the '104 patent, because that is the data that is "obtained" or "thereafter used" in the asserted claims. RT 3311:23-3312:19 (Mitchell); RT 3759:12-3760:23 (McFadden); RT 3954:12-18 (August); 3958:1-3959:4 (August); TX 4015, 7:12-13, 12:16-18, 12:30-31, 12:44-45. Actual data in a data object is what is identified by an exemplary symbolic reference ("y") in the specification. TX 4015, 1:65-67 ("[A]n *instruction that accesses or fetches y*, such as the Load instruction 14' illustrated in FIG. 1, references the variable y by the symbolic name 'y'.") (emphasis added); Fig. 1B (illustrating "data object" containing actual values 23 and 17). The experts agreed the purpose of the LOAD instruction described in the patent is to obtain the value from the data object. RT 3298:20-24 (Mitchell); RT 3960:25-3961:6 (August). The symbolic reference *is to the data obtained*, not some other information, such as information used to resolve the symbolic reference (which is not even illustrated in the figure).

Under the Court's claim construction, a symbolic reference "identifies data." In Android, actual field data in an instance object is "data," as Google's expert testified. RT 4002:5-16 (August). The actual field data in an instance object is "data" in the Court's claim construction, and the field index is the symbolic reference that identifies that data, by a name other than the numeric memory location of the data. Thus the jury's verdict was not reasonable.

3. Conversion of instruction stream indices to numeric memory locations confirms that the indices are symbolic references

That JMOL of infringement should be granted is confirmed by the fact that Dalvik's field indices are resolved to numeric references. If they were numeric references and not symbolic references, there would be no need to convert them to numeric references. But the indices are resolved. Google engineer McFadden, who wrote the Dalvik source code at issue, confirmed this at trial. RT 3236:6-11 (McFadden). Mr. McFadden's source code comments also establish that the Dalvik resolving functions convert an index contained in the instruction stream into a pointer:

> When a class, method, field, or string constant is referred to from Dalvik bytecode, *the reference takes the form of an integer index value*. This value indexes into an array of type_id_item, method_id_item, field_id_item, or string_id_item in the DEX file. The first three themselves contain (directly or indirectly) indexes to strings that *the resolver uses to convert the instruction stream index into a pointer to the appropriate object or struct*.

TX 46.14 at 1 (emphases added). Mr. McFadden confirmed this was an accurate description, and that if the instruction stream index were the numeric memory location, it would already be a pointer and there would be no need to convert it. RT 3234:22-3235:13, 3236:12-19 (McFadden).

B. Android dexopt infringes claims 27 and 29 of the '104 patent

Oracle is also entitled to JMOL on infringement of '104 patent claims 27 and 29 by Android's dexopt. Google's engineers testified that dexopt resolves symbolic references into numerical references. *See, e.g.*, RT 3769:8-12 (McFadden). There were only two disputed issues regarding infringement: whether Dalvik dexopt bytecode instructions contain symbolic references and whether dexopt resolves symbolic references dynamically rather than statically. *See, e.g.*, RT 3841:2-19 (August). The first issue is the same as that with respect to Android's Resolve.c and should be resolved in Oracle's favor as discussed above. With respect to the second issue, the evidence at trial showed that dexopt resolves symbolic references dynamically, not statically.

Mr. McFadden admitted that the resolution process depends on the conditions actually existing on the handset. RT 3769:13-17 (McFadden); *see also* RT 3255:20-25 (McFadden) (admitting need to run dexopt when performing system update because memory layout could change). Dr. Mitchell agreed. RT 3330:24-3331:21 (discussing McFadden testimony). That is sufficient under the ordinary meaning of "dynamic."

Undisputed testimony established that dexopt is performed with a running Dalvik virtual machine. RT 3580:21-23 (Bornstein). When asked whether "dexopt process[es] the dex files when the Dalvik Virtual machine is running," Google expert David August responded, "Sometimes." RT 3988:14-3989:23 (August). That dexopt runs at "runtime" is another sufficient, although not necessary, basis on which to show dynamic reference resolution. Dexopt must process dex files while the Dalvik Virtual Machine is running because it needs information only available at runtime, as Google's internal documentation confirmed. TX 105 at 2-3. Responding to a customer question asking why dexopt had to run at runtime rather than compile time, a Google engineer said it was "normal behavior" and quoted from that documentation to provide an explanation as to "why some of these optimizations can only be performed at runtime." TX 1094.

The ordinary meaning of "dynamic" does not require "at runtime." Mr. McFadden admitted that dexopt is dynamic if "dynamic" means "depending on conditions on the handset which can change from time to time." *See, e.g.*, RT 3769:23-3770:1 (McFadden). But even if "dynamic" did require resolution "at runtime," no reasonable jury could find that dexopt did not run at runtime. The Court should grant JMOL in Oracle's favor.

III. GOOGLE INFRINGED THE ASSERTED CLAIMS OF THE '520 PATENT

Oracle proved Google's dx tool infringes Claims 1 and 20 of the '520 patent.

As stated in the patent handouts, Google concedes all steps of Claims 1 and 20 are performed except the "simulating execution" step. TX 1106 at 7-8. But the indisputable evidence is that "simulating execution" *is* performed by the dx tool; indeed the author of the dx tool *expressly described* it as such in the source code comments. RT 3547:20-21 (Bornstein); TX 46.16 at line 37 ("Class which knows how to simulate the effects of executing bytecode").

Google's code also demonstrates that the dx tool performs these "simulating execution" steps. A code file called "Simulator.java" within the dx tool simulates execution of Java bytecodes to convert them to Dalvik bytecodes. The engineer comments in the code state that Simulator.java is a class designed to "simulate the effects of executing bytecode." TX 46.16 at lines 37-43, 86-105. The file calls upon the parseInstruction and parseNewarray methods to assist with understanding the instructions. TX 46.16 at line 99; TX 46.17 at lines 211, 887; *see also* RT 3341:17-3344:7 (Mitchell). As a result of Simulator.java and the methods it invokes, the bytecode instructions are examined without being executed, their static initialization is determined, and a shorter "fast instruction" is generated to replace the long list of bytecode instructions. *Id.* This precisely matches the "simulating execution" step of the asserted claims.

Google's expert, Dr. Parr, conceded the dx tool does "identify the static initialization of the array" by examining the "bytecodes of the clinit method against a memory" and "without executing the bytecodes." RT 3793:2-5, 3807:10-14, 3820:12-22, 3821:16-23, 3822:17-3823:13 (Parr). With those concessions, he acknowledged the sole remaining issue was whether the dx tool process for identifying static initializations could be characterized as "simulating execution."

To avoid the effect of these admissions, Google imported additional, non-existent limitations into the claims. First, Dr. Parr claimed the dx tool cannot be simulating execution of bytecodes because it does not manipulate a stack to determine static initializations of arrays. RT 3794:15-3795:21, 3801:19-21 (Parr). But as Dr. Parr conceded, the asserted claims do not mention stack manipulation. TX 4011, 9:47-62, 12:3-7; RT 3794:20-23 (Parr); s*ee also* RT 4032:23-4033:8 (Mitchell). Nor was the term "simulating execution" construed to require stack manipulation. "Stack manipulation" is an express limitation of dependent Claim 3, establishing that the limitation is not a requirement of "simulating execution" in independent Claim 1.

Second, Dr. Parr argued that the dx tool identifies static initializations through pattern matching, which he contends is distinguishable from "simulating execution" of bytecodes. RT 3798:22-3799:3. But there is nothing in the claim language or patent specification stating "simulating execution" cannot be achieved through pattern matching. Indeed, because Android's pattern matching determines the effect of executing the static initialization bytecodes *without actually executing them*, no reasonable jury could find the dx tool did not simulate execution.

The Court denied Oracle's Rule 50(a) motion for JMOL, limiting "simulate execution" to "manipulation of a stack by pushing, popping, and replacing values from the top of an operand stack." ECF No. 1201 at 10. In doing so, the Court limited the scope of Claims 1 and 20 to a disclosed embodiment, in contravention of the claim language. Under the ordinary meaning of "simulate execution," Oracle is entitled to JMOL.

IV. GOOGLE'S EQUITABLE DEFENSES FAIL

The Court has already granted judgment on Google's defenses of waiver and implied license. ECF No. 1203. It should also grant judgment as to equitable estoppel and laches.

Oracle incorporates by reference its proposed findings of fact and conclusions of law and prior Rule 50(a) briefing on Google's equitable defenses. *See* ECF No. 1049 at 11-26, 30-35; ECF No. 1081 at 16-37, 55-70.

A. Google Has Not Met Its Burden Of Proving Equitable Estoppel

Google failed to prove any of the elements of equitable estoppel. In particular, Google has no credible claim that it relied on Sun/Oracle's conduct to its detriment or that its reliance was reasonable. The jury so advised in its Phase I ruling. ECF No. 1089 ¶ 4.B. Overwhelming evidence at trial showed Google was aware that Sun had copyrighted its Java source code and API specifications and had patents that covered its virtual machine technology and that it faced potential legal action by Sun in connection with Android. *See, e.g.*, RT 695:11-697:19 (Reinhold); RT 756:9-18 (Bloch); RT 951:8-953:9 (Swetland); RT 983:4-15 (Lee); RT 1541:3-7 (Schmidt); RT 1356:6-19, 1689:19-25, 3204:6-3205:3 (Rubin); RT 2993:4-24 (Lindholm); TX 18; TX 25 at 389; TX 149; TX 273; TX 405; TX 610.1 at 1; TX 610.2; TX 980 at 6; TX 1029; TX 1051 at 1; TX 2347; ECF No. 1049 ¶¶ 62-65, 130, 132; ECF No. 1081 ¶ 66. Google decided on its Android development path and implemented its infringing technology regardless of any Sun or Oracle statements, actions, or inactions, and Google has never proven otherwise. TX 1029; ECF No. 1049 ¶¶ 62-65, 96, 98, 114, 117. JMOL against Google is warranted.

B. Google Has Not Shown Laches Bars Oracle's Infringement Claims

Google has not produced evidence to support a laches defense. To prove laches, Google must show that (1) Oracle/Sun unreasonably delayed filing the lawsuit; (2) the delay was inexcusable, and (3) Google suffered material prejudice due to Oracle/Sun's delay. *Danjaq LLC v. Sony Corp.*, 263 F.3d 942, 952-57 (9th Cir. 2001). Oracle sued Google on August 12, 2010, less than two years after Google first made its Android source code available to the public (in October 2008) and less than three years after Google first released the Android APIs (in November 2007). RT 1041:14-16 (Morrill); RT 1546:14-16 (Schmidt); RT 1702:22-1704:9 (Rubin); RT 1719:10-18 (Rubin). As a result, there is no presumption of laches, and "the burden is upon the defendant to show that the delay was unexcused and that the defendant suffered injury as a result of the delay." *Carpet Seaming Tape Licensing Corp. v. Best Seam, Inc.*, 694 F.2d 570, 580 (9th Cir. 1982); *A.C. Aukerman Co. v. R.L. Chaides Const. Co.*, 960 F.2d 1020, 1038 (Fed.

Cir. 1992) (*en banc*). In 2008 and 2009, Sun engaged in licensing discussions with Google, making any delay reasonable and excusable. *See, e.g.*, RT 1071:23-1073:18 (Cizek); TX 1002; TX 1029; *In re Katz Interactive Call Processing Patent Litig.*, 712 F. Supp. 2d 1080, 1110 (C.D. Cal. 2010); *Aukerman*, 960 F.2d at 1033. Google also made no showing of material prejudice.

Furthermore, laches is not available in a case of willful infringement. *See id.* at 1032.

V. THE COURT SHOULD GRANT JMOL ON GOOGLE'S ALTERNATIVE DEFENSES TO PATENT INFRINGEMENT

Oracle moved for JMOL as to Google's Sixth (patent misuse), Eighth (use by the U.S.) and Nineteenth (unclean hands) defenses, as well as Google's defense of express license. ECF No. 1168 at 24. Google never presented these defenses at trial and did not oppose the motion. ECF No. 1169 at 18-19. Accordingly, the Court should grant JMOL in Oracle's favor.

VI. IN THE ALTERNATIVE, ORACLE IS ENTITLED TO A NEW TRIAL

In the alternative, if the Court declines Oracle's request for JMOL, Oracle requests a new trial on the issues of: (i) Google's copying of comments; (ii) Google's copying from Java documentation into Android documentation; (iii) Google's creation of an unauthorized derivative work; (iv) copyrightability of the 37 Java API packages; and (v) Google's infringement of the '104 and '520 patents. In addition, because the jury did not reach a verdict on fair use, Oracle is entitled to a new trial on that issue if the Court's copyrightability order is reversed.

Following a jury trial, a court may grant a new trial "for any reason for which a new trial has heretofore been granted in an action at law in federal court." Fed. R. Civ. P. 59(a)(1)(A). "Historically recognized grounds include, but are not limited to, claims 'that the verdict is against the weight of the evidence, that the damages are excessive, or that, for other reasons, the trial was not fair to the party moving.'" *Molski v. M.J. Cable, Inc.*, 481 F.3d 724, 729 (9th Cir. 2007) (citation omitted). "[E]rroneous jury instructions, as well as the failure to give adequate instructions, are also bases for a new trial." *Murphy v. City of Long Beach*, 914 F.2d 183, 187 (9th Cir. 1990). The cumulative prejudice from multiple errors may warrant a new trial. *See, e.g., United States v. Necoechea*, 986 F.2d 1273, 1282 (9th Cir. 1993). A new trial may be held on issues not tried to the jury if it could be obtained under similar circumstances in a jury action.

12 James W. Moore et al., *Moore's Federal Practice* § 59.13[3][a] (3d ed. 2012).

Oracle is entitled to a new trial for all the same reasons it is entitled to JMOL. These reasons are described in detail above and are incorporated by reference here. For the jury issues these include, but are not limited to: (1) the jury's finding of non-infringement on copied comments, documentation and the '104 and '520 patents is against the clear weight of the evidence; (2) the Court erroneously instructed the jury (a) to compare only the English language descriptions of the Java and Android specifications, and (b) to apply a virtual identity standard; (3) the Court erred in not submitting to the jury the issue of Google's creation of a derivative work from the Java documentation.

Oracle is also entitled to a new trial as to infringement of the '104 patent based on the Court's failure to give a curative instruction and its erroneous response to the juror's question on May 22 at 10:35 a.m. In that response the Court stated, among other things, that the reference "is either going to be a numeric reference or it's going to be a symbolic reference" but "can't be both." *See* RT 4352:8-4354:8; ECF No. 1189. Neither the '104 patent nor the Court's claim construction order contain such a limitation. The jury returned its verdict shortly thereafter.

Oracle is entitled to a new trial on the Court's copyrightability order on the grounds that it would cause manifest injustice and contains manifest errors of law and fact. These errors include, but are not limited to: (1) the finding that to carry out a given function method declarations must be identical; (2) the conclusion that a method specification is an idea and its implementation is expression; (3) the conclusion that the Java APIs are an uncopyrightable method of operation or system; (4) the finding that a user must make use of Oracle's java.lang, java.io and java.util packages to make worthwhile use of the language; (5) the finding that Google replicated what was necessary to achieve a degree of interoperability, but no more.

Oracle reserves all other grounds as to which a new trial may be granted on appeal.

CONCLUSION

For the foregoing reasons, Oracle is entitled to judgment in its favor as set forth above. In the alternative, the Court should grant Oracle's motion for a new trial.

Dated: June 20, 2012

MICHAEL A. JACOBS
MARC DAVID PETERS
DANIEL P. MUINO
MORRISON & FOERSTER LLP

By: <u>*/s/ Michael A. Jacobs*</u>
Michael A. Jacobs

Attorneys for Plaintiff
ORACLE AMERICA, INC.